Dept. Computer Science	Operating Systems
Vrije Universiteit	06.06.2007

1a Explain the difference between seeing an operating system as a **virtual machine** or a **resource manager**. *5pt*

As a virtual machine, an operating system provides an abstraction over the hardware by means of, for example, system calls. In this way, it provides a convenient way to program a machine without the need to know about hardware details. As a resource manager, it allows multiple processes to share the various resources such as CPU, storage, and network. Its role of manager consists of protecting those resources against simultaneous access, protecting processes against each other, supports fair sharing of those resources, and accounts processes for resource usage.

1b Cygwin is described as being a Linux-like environment for Windows. An important component is "a library which acts as a Linux API emulation layer providing substantial Linux API functionality." Does this mean that you can, in principle, run any Linux application in Cygwin? Explain your answer! *5pt*

The description is a bit vague, but it should be clear that Cygwin essentially provides the same system calls as Linux would normally do. This means that if you have the source code *of a Linux application, you should in principle be able to compile it, link it with the Cygwin library, and then run it as a regular Windows application. Crucial in your answer is that you need to compile applications before being able to execute them in Cygwin. In this sense, Cygwin is* not *the same as a virtual machine.*

1c What is in your view the main difference between executing a Cygwin application and executing a Java application? Be sure to explain your answer. *5pt*

The difference is subtle, yet important. A Cygwin application is a Windows application of which the source code makes use of the Cygwin API, which consists of Linux-like system calls. Note that Cygwin applications are written in C. The net result is a native Windows applicaton. A Java application, on the other hand, is translated into an intermediate set of (so-called bytecode) instructions, which are then subsequently interpreted by the Java virtual machine. This means that compiled Java applications can be ported to other operating systems, provided there is a JVM available on that platform.

2a Give a solution to the producer-consumer problem using **(regular) semaphores**. *5pt*

```
#define N 100
typedef int semaphore;
semaphore mutex = 1;
semaphore empty = N;
semaphore full  = 0;

void producer(){
  int item;
  while(TRUE){
    item = produce_item();
    down(&empty);
    down(&mutex);
    insert_item(item);
    up  (&mutex);
    up  (&full);
  }
}

void consumer(){
  int item;
  while(TRUE){
    down(&full);
    down(&mutex);
    item = remove_item(item);
    up  (&mutex);
    up  (&empty);
    consume_item(item);
  }
}
```

2b Give a solution to the producer-consumer problem using **binary semaphores**. *10pt*

```
#define N 100
typedef int semaphore;
semaphore mutex = 1;
semaphore produced = 0;
semaphore consumed = 0;
int empty = N;
int full  = 0;

void producer(){
  int item;
  while(TRUE){
    item = produce_item();
    done = false;
    while( !done ){
      down(&mutex);
      if( empty > 0 ){
        insert_item(item);
        empty = empty - 1;
        full  = full + 1;
        done  = true;
        up(&produced);
        up(&mutex);
      }
      else{
        up(&mutex);
        down(&consumed)
      }
    }
  }
}
```

```
void consumer(){
  int item;
  while(TRUE){
    done = false;
    while( !done ){
      down(&mutex);
      if( full > 0 ){
        item  = remove_item(item);
        full  = full - 1;
        empty = empty + 1;
        done  = true;
        up(&consumed);
        up(&mutex);
      }
      else{
        up(&mutex);
        down(&produced);
      }
    }
    consume_item(item);
  }
}
```

3a Describe the flow of control from the moment that a device generates an interrupt to the moment that the first instructions of an interrupt handler are executed. It is important to be precise. *5pt*

Your answer should at the very least contain the following elements. (1) the interrupt controller passes a reference to interrupt vector to the CPU. (2) the CPU puts essential information on the interrupted program on the current stack, e.g., program counter. (3) CPU reads the start address of the interrupt handler from the interrupt vector (which is part of the interrupt descriptor table), and loads that into the program counter.

3b Disabling interrupts used to be a standard technique for protecting critical sections in operating systems. Will this technique also work for an operating system designed for a multicore processor? *5pt*

No: disabling interrupts lets a processor continue execution without indeed being interrupted. That means that a single thread of control is executing. However, with multicore processors we can easily have multiple threads executing at the same time, and disabling interrupts is not going to prevent another thread from accessing shared data.

4a What are the sufficient and necessary conditions for deadlock to occur? *5pt*

(1) an allocated resource cannot be taken away. (2) a process is allowed to request a resource while holding another one. (3) each resource is assigned to at most one process, (4) there is a cirular wait: two or more processes are waiting for each other to release a resource that the other wants.

4b Consider the following allocation of resources R1, R2, R3, and R4. Show that this is a safe state. *5pt*

Process	R1	R2	R3	R4
A	4	1	0	1
B	0	2	0	0
C	1	0	1	0
D	1	0	0	1
E	0	0	1	0

Allocated resources

Process	R1	R2	R3	R4
A	1	1	0	0
B	0	1	1	2
C	3	1	0	0
D	0	0	1	1
E	2	1	1	0

Resources still needed

R1	R2	R3	R4
7	4	2	2

Originally available

The state is safe, as we can complete the allocation in the order A - C - D - B - E.

5a Explain the principle of translating an address for a paged virtual memory system. *5pt*

Such an address consists of at least two parts: an index into a page table, and an offset into a page. The index is used to lookup which page frame is associated with the address, if any. If there is no associated page frame, this means that the data is stored on disk and needs to be swapped in. At that point, a page frame is selected, and, if necessary, written to disk. That frame is then used to be filled with the data associated with original address, after wich the page's address is stored in the process' page table. This will then allow the MMU to locate the page frame, and subsequently use the offset to find the actual location in main memory.

5b Least-recently used (LRU) page frames are often best to be evicted from memory, if there is a need to do so. Describe a technique that approximates keeping track of such page frames. *5pt*

The trick is to keep a bit string variable, per page frame, indicating whether it was recently referenced. This variable is initialized to contain only zeroes. At every clock tick (or time unit for deciding on evicting pages), the value of frame's reference bit is shifted from the left into the bit string. As a consequence, when interpreting the bit string as a binary number, the least-recently used page generally have the lowest number (but not always - see book).

5c Consider two identical computers *A* and *B*, both supporting virtual memory, which are connected through a network. Sketch a simple (perhaps naïve) scheme such that processes running on *A* and *B* can share memory. *5pt*

When a process on A addresses a page that is not associated with a page frame, A simply forwards the request to B to see whether that address is associated with a page frame. If so, it ***moves*** *the page frame to A as if fetching it from disk in the case of normal virtual memory. In other words, the page frame at B must be invalidated. If neither A nor B has an associated page frame, A can allocate a page frame and fill it with data from (shared) disk.*

6a Consider the standard layout of a UNIX file system. Keeping track of available disk blocks and i-nodes is done through bit maps. How do you determine the maximum number of files that a file system can handle? *5pt*

The file-system layout includes disk blocks for i-nodes. If there are N disk blocks, each of size d bytes, then with an i-node size of i, the maximum number of files is $N \times d/i$*. In this case, we can assume that the index for keeping track of available disk blocks is larger than N bits. Note that simply stating that the size of the i-node index determines the maximum number of files is wrong.*

6b Consider again a UNIX file system. How do you determine the maximum file size? *5pt*

In this case, you need to concentrate on the structure of the i-nodes. For simplicity, let's assume it supports single-indirect and double-indirect disk blocks. With a disk block size of d bytes, and disk block addresses being 4 bytes, we can store $d/4$ *addresses per block. The maximum file size is then approximately* $(d/4)^2 \times d$*. Note that in this case, you do need to say something about the initially available disk blocks: for USB sticks, the maximum file size is determined by the available space.*

7a There are various tools available that will allow the so-called auto-play of a CD ISO image in Windows. Often, you need to first create a (virtual) software CD drive, and then mount the image on that drive. What does the creation of the software CD drive actually entail? What would be the equivalent action under UNIX? *5pt*

Creating a software drive is nothing but creating the means such that other tools can access data as if it where placed on a real drive. The equivalent in UNIX is actually creating a mount point.

7b What does mounting the CD ISO image actually establish in Windows? *5pt*

By mounting the ISO image, we have now completed the simulation of having a CD available for reading by various programs, including the one for auto-playing the image. In this case, the auto-play program knows where to access the CD (the drive is known), and will search for a file in the associated file system giving it the right information for continuation (i.e., it will look for "autorun.inf").

7c Is it possible in this way to also boot and install a new operating system? *5pt*

It really depends: when booting from such an ISO image, the first thing that will generally happen is that you boot into a simple OS that will steer the rest of the OS installation. To that end, it will need to copy files and such from the software drive, which by then has been lost. On the other hand, if you have a special installer that runs under the current OS, it should be theoretically possible to first copy everything that is needed to install the new operating system to a safe place on disk, change the boot record on disk, and then request the user to reboot.

Grading: *The final grade is calculated by adding the scores per question (maximum: 90 points), and adding 10 bonus points. The maximum total is therefore 100 points.*